WHAT IS CLAIMED IS:

1.   A method for interfacing a genetic search algorithm to the web environment of the Internet, comprising:

defining data elements, attributes and rules for use thereof for an extensible markup language;

storing the extensible markup language in a document type definition file;

describing data in a document in a hierarchical format utilizing the stored extensible markup language; and

importing the data in the document to a genetic algorithm to define a data string or an individual in a population of points.

2.   A method for interfacing a genetic search algorithm as in Claim 1 wherein defining data elements, attributes and rules comprises defining a tree-based crossover operator.

3.   A method for interfacing a genetic search algorithm as in Claim 1 wherein defining data elements, attributes and rules comprises defining a mutation operator.

4.   A method for interfacing a genetic search algorithm as in Claim 3 further comprising importing elements from the document definition file as nodes of the stored extensible markup language.

5. A method for interfacing a genetic search algorithm as in Claim 4 wherein defining a mutation operator comprises performing the mutation operator either at an internal or external node.

6. A method for interfacing a genetic search algorithm as in Claim 1 wherein defining data elements, attributes and rules comprises defining a permutation operator, an editing operator, an encapsulation operator, and a dissemination operator.

7. A method for interfacing an extensible markup language genetic search algorithm to the web environment of the Internet, comprising:

    providing a genetic algorithm parameter document containing input data;

    creating a population of individuals from the input data;

    randomly generating documents conforming to the genetic algorithm individual DTD; and

    providing utilities for performing crossover operations and/or mutation operations.

8. A method for interfacing a genetic search algorithm as in Claim 7 wherein providing a parameter document comprises providing constraints for the created individuals.

9.   A   method   for   interfacing   a   genetic   search
algorithm as in Claim 8 wherein providing constraints for
the created individuals comprises providing one or more
constraints   from   the   group   comprising: uniqueness,
5   completeness and maximum and minimum values for nodes
that contain text data.

10.   A   method   for   interfacing   a   genetic   search
algorithm as in Claim 7 wherein creating a population of
10   individuals comprises creating programs organized into a
tree structure.

11.   A   method   for   interfacing   a   genetic   search
algorithm as in Claim 10 wherein providing utilities
15   comprises providing a crossover operator for structuring
nodes of the tree structure.

12.   A   method   for   interfacing   a   genetic   search
algorithm as in Claim 10 wherein providing utilities
20   comprises providing a mutation operator for performing on
either internal or external nodes of the tree structure.

13.   A   method   for   interfacing   a   genetic   search
algorithm as in Claim 7 wherein providing utilities
25   comprises providing a bit-wise crossover operator and/or
a tree-node crossover operator.

14. A method for interfacing an extensible markup language genetic search algorithm to the web environment of the Internet, comprising:

providing a genetic algorithm parameter document containing input data;

creating a population of individuals from the input data;

randomly generating documents conforming to the genetic algorithm individual DTD; and

providing utilities for performing one or more of the operators from the group comprising: crossover, mutation, permutation, editing, encapsulation, and dissemination.

15. A method for interfacing a genetic search algorithm as in Claim 14 wherein providing utilities comprises providing a permutation operator for reordering the characters found between two selected points of a single individual.

16. A method for interfacing a genetic search algorithm as in Claim 14 wherein providing utilities comprises providing an editing operator for recursively applying a pre-established set of editing rules to each individual member of the population.

17. A method for interfacing a genetic search algorithm as in Claim 14 wherein creating a population of individuals comprises creating programs organized into a tree structure.

18. A method for interfacing a genetic search algorithm as in Claim 17 wherein providing utilities comprises providing an encapsulation operator for identifying and naming potentially useful sub trees to enable referencing for later use.

19. A method for interfacing a genetic search algorithm as in Claim 14 wherein providing utilities comprising providing a dissemination operator for ridding the population of individuals of low-fitness individuals in cases where the population of individuals becomes skewed and has a high proportion of such individuals.

20. A method for interfacing an extensible markup language genetic search algorithm to the web environment of the Internet, comprising:

providing a genetic algorithm parameter document containing input data;

providing an XSL style sheet containing instruction for the genetic algorithm to construct a population of individuals;

creating a population of individuals from the input data in accordance with the instructions from the XSL style sheet;

randomly generating documents conforming to the genetic algorithm individual DTD;

providing a fitness function for the individuals of the created population; and

providing utilities for performing one or more of the operators selected from the group comprising: crossover, mutation, permutation, editing, encapsulation, and dissemination.

21. A method for interfacing a genetic search algorithm as in Claim 20 wherein providing a parameter document comprises providing constraints for the created individuals.

22. A method for interfacing a genetic search algorithm as in Claim 21 wherein providing constraints comprises providing one or more constraints from the group comprising: uniqueness, completeness, and maximum and minimum values for nodes that contain text data.

23. A method for interfacing a genetic search algorithm as in Claim 20 wherein creating a population of individuals comprises creating programs organized into a tree structure.

24. A method for interfacing a genetic search algorithm as in Claim 23 wherein providing utilities comprises providing a bit-wise crossover operator and/or a tree-node crossover operator.